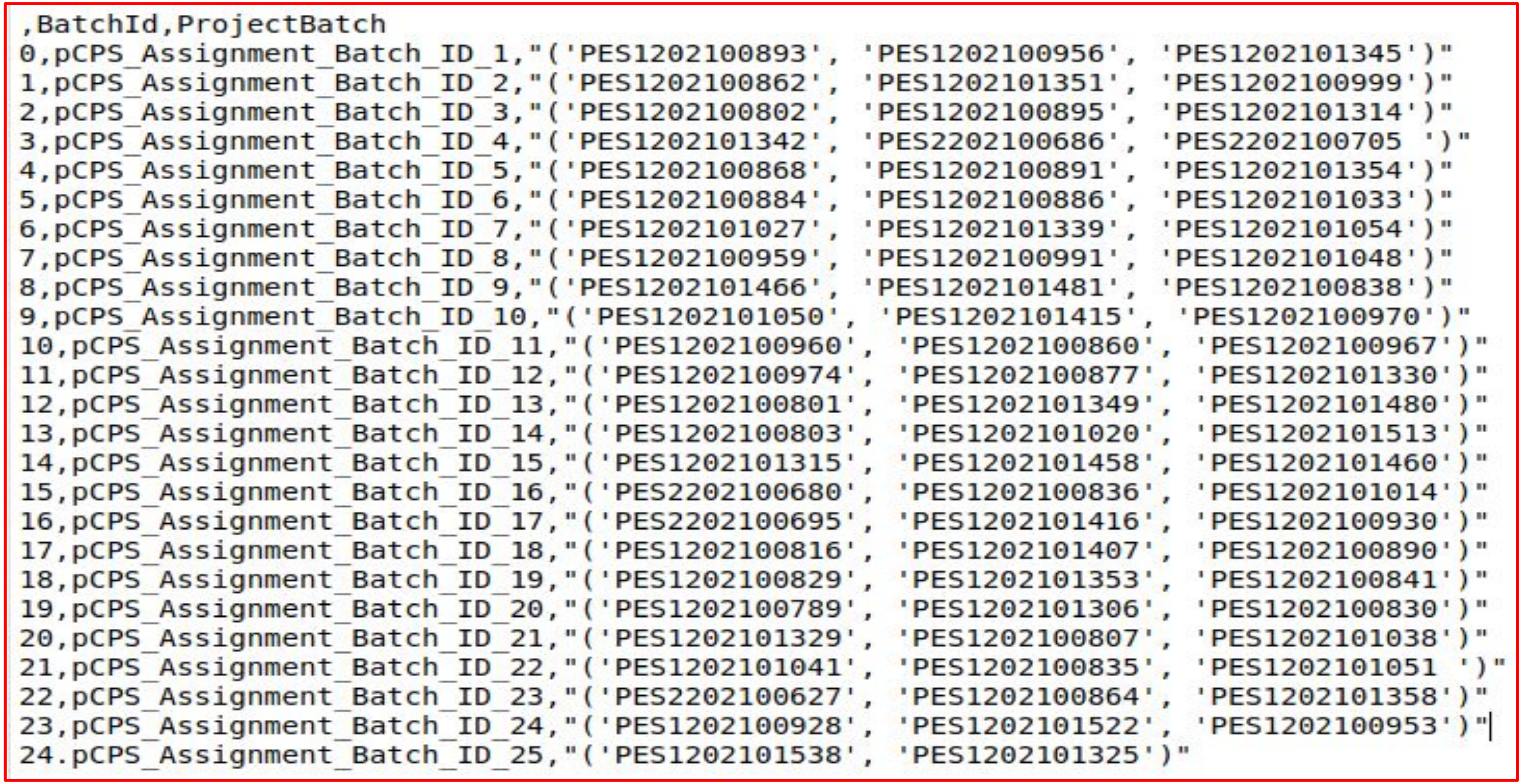
python for Computational Problem Solving - pCPS - String_Processing Lecture Slides - Class #23_#24

Nitin V Pujari
Faculty, Computer Science
Dean - IQAC, PES University

pCPS Assignment Batches



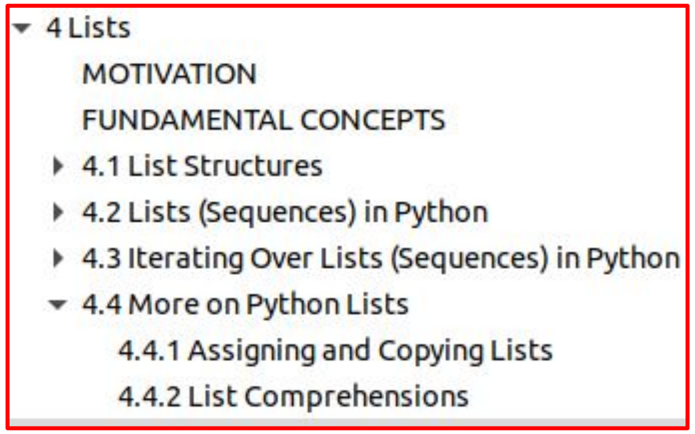
```
,BatchId,ProjectBatch
0,pCPS_Assignment_Batch_ID_1,"('PES1202100893', 'PES1202100956', 'PES1202101345')"
1,pCPS_Assignment_Batch_ID_2,"('PES1202100862', 'PES1202101351', 'PES1202100999')"
2,pCPS_Assignment_Batch_ID_3,"('PES1202100802', 'PES1202100895', 'PES1202101314')"
3,pCPS_Assignment_Batch_ID_4,"('PES1202101342', 'PES2202100686', 'PES2202100705 ')"
4,pCPS_Assignment_Batch_ID_5,"('PES1202100868', 'PES1202100891', 'PES1202101354')"
5,pCPS_Assignment_Batch_ID_6,"('PES1202100884', 'PES1202100886', 'PES1202101033')"
6,pCPS_Assignment_Batch_ID_7,"('PES1202101027', 'PES1202101339', 'PES1202101054')"
7,pCPS_Assignment_Batch_ID_8,"('PES1202100959', 'PES1202100991', 'PES1202101048')"
8,pCPS_Assignment_Batch_ID_9,"('PES1202101466', 'PES1202101481', 'PES1202100838')"
9,pCPS_Assignment_Batch_ID_10,"('PES1202101050', 'PES1202101415', 'PES1202100970')"
10,pCPS_Assignment_Batch_ID_11,"('PES1202100960', 'PES1202100860', 'PES1202100967')"
11,pCPS_Assignment_Batch_ID_12,"('PES1202100974', 'PES1202100877', 'PES1202101330')"
12,pCPS_Assignment_Batch_ID_13,"('PES1202100801', 'PES1202101349', 'PES1202101480')"
13,pCPS_Assignment_Batch_ID_14,"('PES1202100803', 'PES1202101020', 'PES1202101513')"
14,pCPS_Assignment_Batch_ID_15,"('PES1202101315', 'PES1202101458', 'PES1202101460')"
15,pCPS_Assignment_Batch_ID_16,"('PES2202100680', 'PES1202100836', 'PES1202101014')"
16,pCPS_Assignment_Batch_ID_17,"('PES2202100695', 'PES1202101416', 'PES1202100930')"
17,pCPS_Assignment_Batch_ID_18,"('PES1202100816', 'PES1202101407', 'PES1202100890')"
18,pCPS_Assignment_Batch_ID_19,"('PES1202100829', 'PES1202101353', 'PES1202100841')"
19,pCPS_Assignment_Batch_ID_20,"('PES1202100789', 'PES1202101306', 'PES1202100830')"
20,pCPS_Assignment_Batch_ID_21,"('PES1202101329', 'PES1202100807', 'PES1202101038')"
21,pCPS_Assignment_Batch_ID_22,"('PES1202101041', 'PES1202100835', 'PES1202101051 ')"
22,pCPS_Assignment_Batch_ID_23,"('PES2202100627', 'PES1202100864', 'PES1202101358')"
23,pCPS_Assignment_Batch_ID_24,"('PES1202100928', 'PES1202101522', 'PES1202100953')"
24.pCPS_Assignment_Batch_ID_25,"('PES1202101538', 'PES1202101325')"
```

python for Computational Problem Solving Syllabus

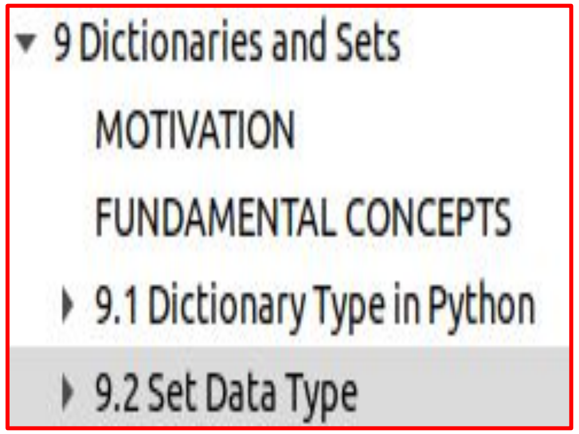
Unit II: Collections & Basics of Functions - 12 Hours

Lists, Tuples , Dictionaries, Sets, Strings and text file manipulation: reading and writing files. Functions : Definition, call.

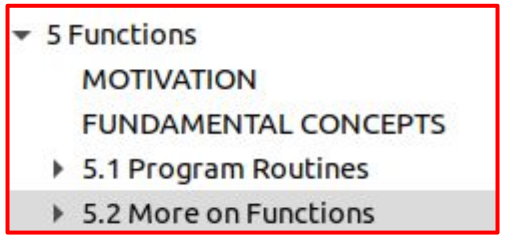
T1: 4.1 – 4.4 - Class #15, #16, #17, #18
T1: 9.1 – 9.2 - Class #19, #20, #21
T1: 5.1-5.2 - Class #25, #26
T1: 8.1, 8.2, 8.3 - Class #22, #23,#24



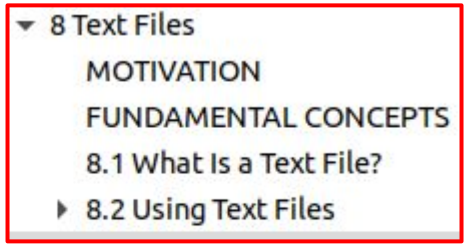
4 Lists
MOTIVATION
FUNDAMENTAL CONCEPTS
4.1 List Structures
4.2 Lists (Sequences) in Python
4.3 Iterating Over Lists (Sequences) in Python
4.4 More on Python Lists
4.4.1 Assigning and Copying Lists
4.4.2 List Comprehensions



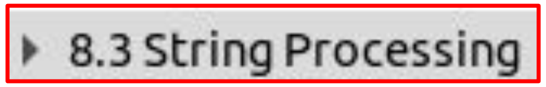
9 Dictionaries and Sets
MOTIVATION
FUNDAMENTAL CONCEPTS
9.1 Dictionary Type in Python
9.2 Set Data Type



5 Functions
MOTIVATION
FUNDAMENTAL CONCEPTS
5.1 Program Routines
5.2 More on Functions



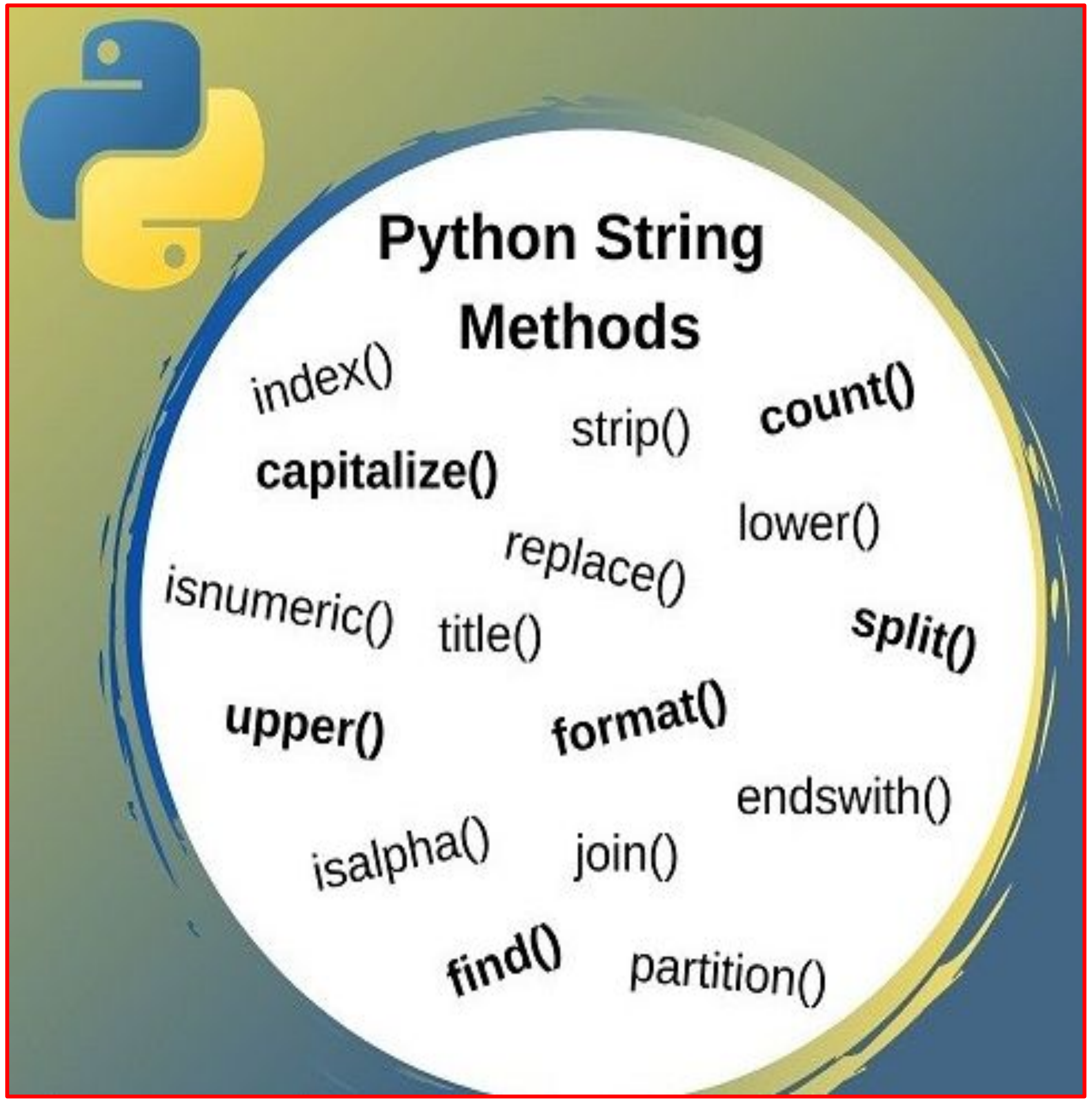
8 Text Files
MOTIVATION
FUNDAMENTAL CONCEPTS
8.1 What Is a Text File?
8.2 Using Text Files



8.3 String Processing

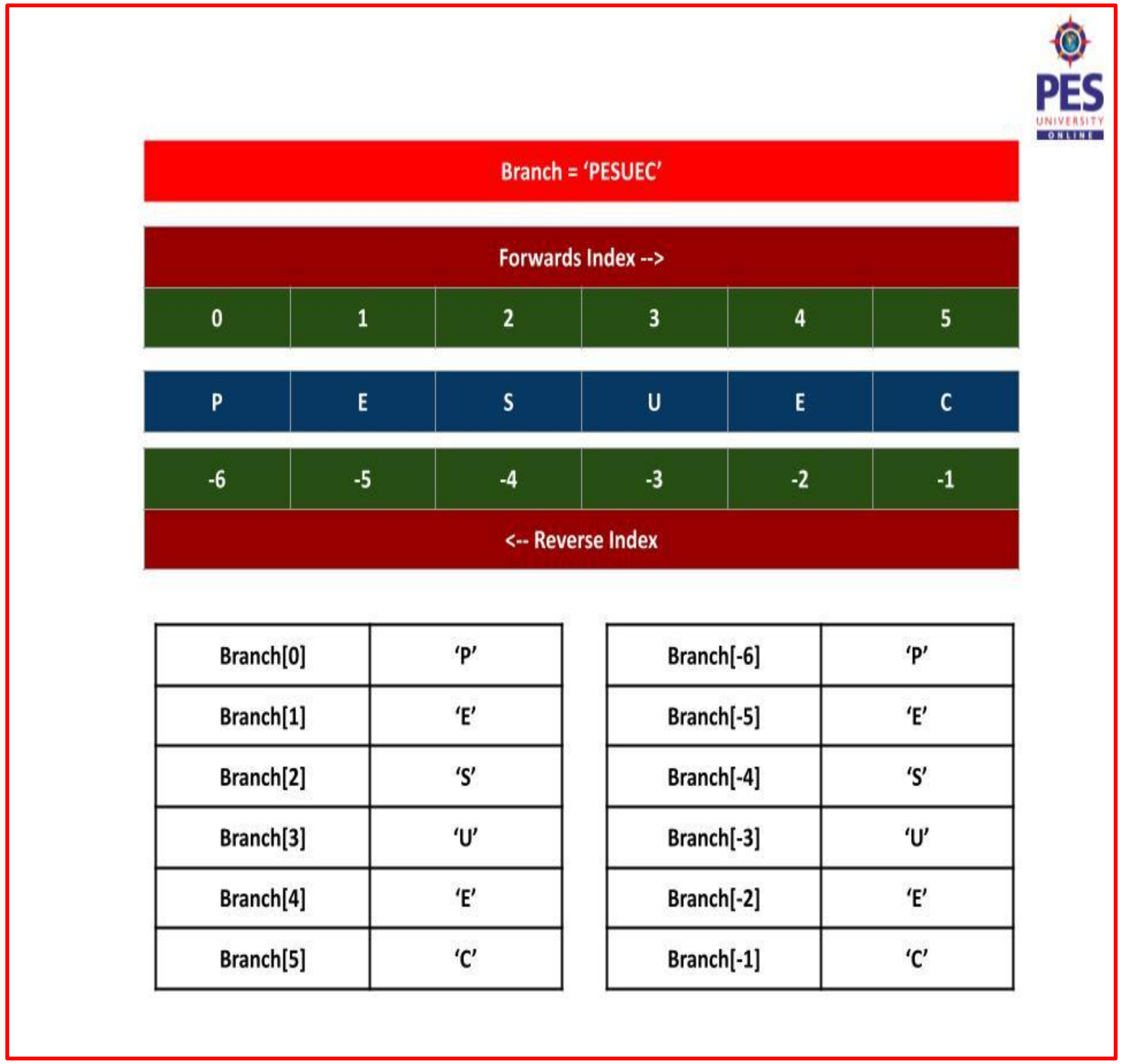
pCPS 8.3 String Processing in python

- The information in a **text file**, as with all information, is most likely going to be searched, analyzed, and/or updated.
- Collectively, the operations performed on strings is called **string processing** .
- **String processing refers** to the **operations** performed on **strings** that allow them to be accessed, analyzed, and updated



pCPS 8.3.1 String Traversal in python

- The **characters** in a string can be easily **traversed**, **without** the use of an **explicit index** variable, using the for <identifier> in string form of the **for** statement



Branch = 'PESUEC'

Forwards Index -->

0	1	2	3	4	5
P	E	S	U	E	C
-6	-5	-4	-3	-2	-1

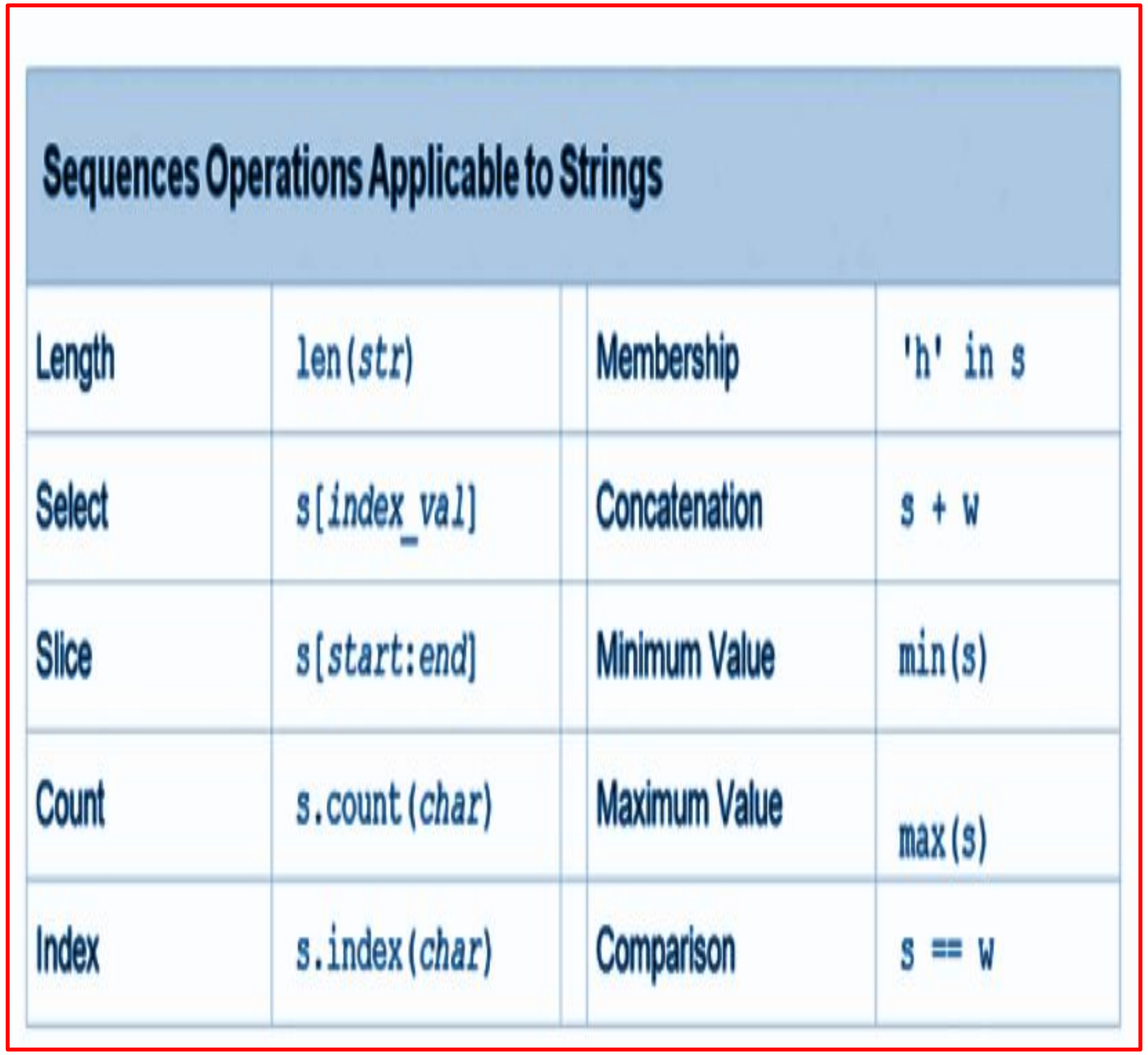
<-- Reverse Index

Branch[0]	'P'
Branch[1]	'E'
Branch[2]	'S'
Branch[3]	'U'
Branch[4]	'E'
Branch[5]	'C'

Branch[-6]	'P'
Branch[-5]	'E'
Branch[-4]	'S'
Branch[-3]	'U'
Branch[-2]	'E'
Branch[-1]	'C'

pCPS 8.3.2 String-Applicable Sequence Operations in python

- **strings** are **immutable**, **sequence**-**modifying** operations are **not applicable** to strings.
- One **cannot** add, delete, or replace characters of a **string**.
- All **string** operations that "**modify**" a **string return** a **new string** that is a **modified** version of the original string .
- The **slice** operator s[**start**:**end**] returns the **substring** starting with index **start**, up to but **not** including index **end**.
- **s.index(chr**) returns the **index** of the **first occurrence** of **chr** in s.

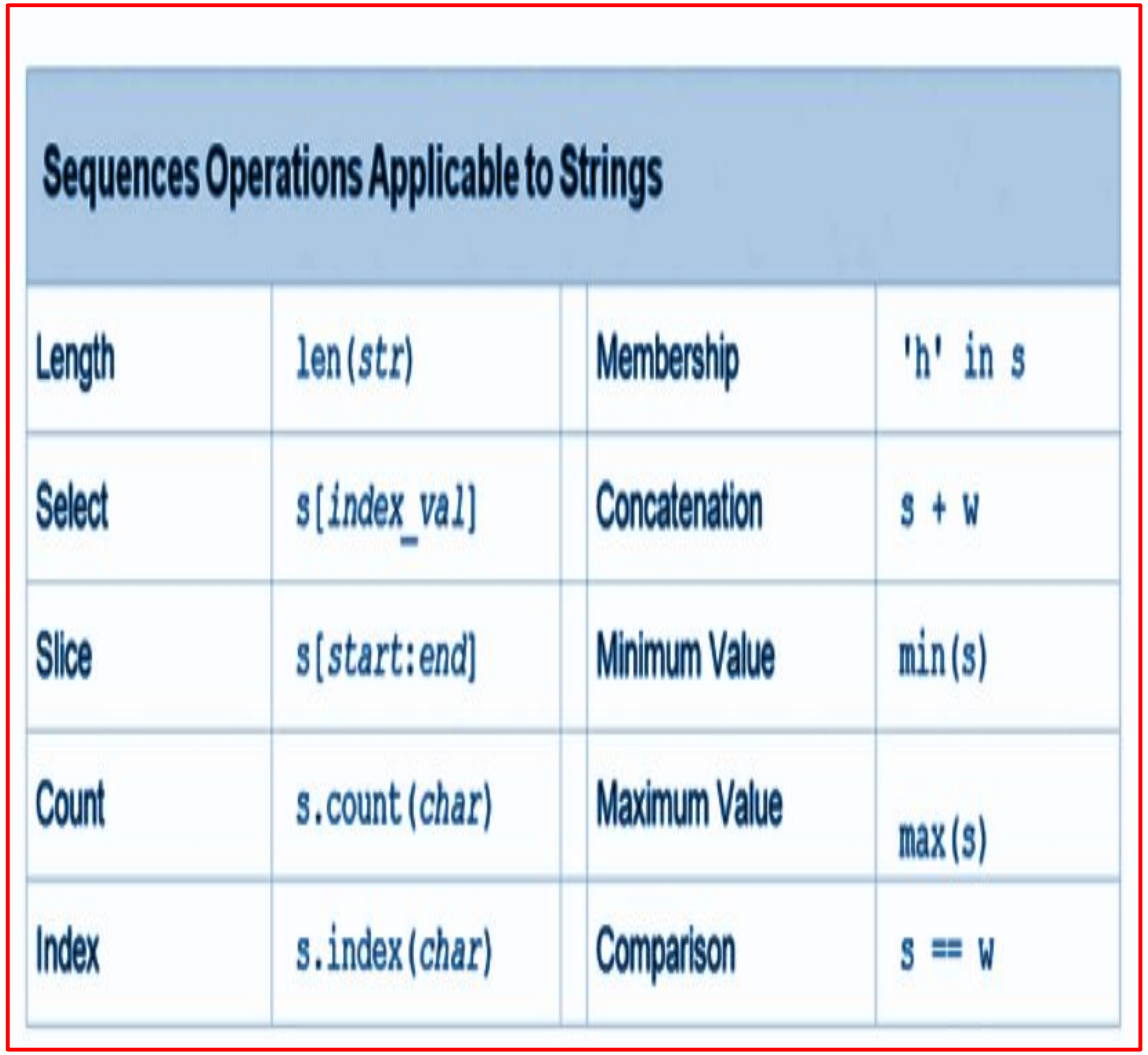


Sequences Operations Applicable to Strings			
Length	len(*str*)	Membership	'h' in s
Select	s[*index_val*]	Concatenation	s + w
Slice	s[*start*:*end*]	Minimum Value	min(s)
Count	s.count(*char*)	Maximum Value	max(s)
Index	s.index(*char*)	Comparison	s == w

pCPS 8.3.2 String-Applicable Sequence Operations in python

- **min** and **max** as applied to strings return the **smallest** and **largest character** respectively based on the underlying **Unicode** encoding.

- For example, all **lowercase** letters are **larger** have a larger **Unicode** value than all **uppercase** letters.

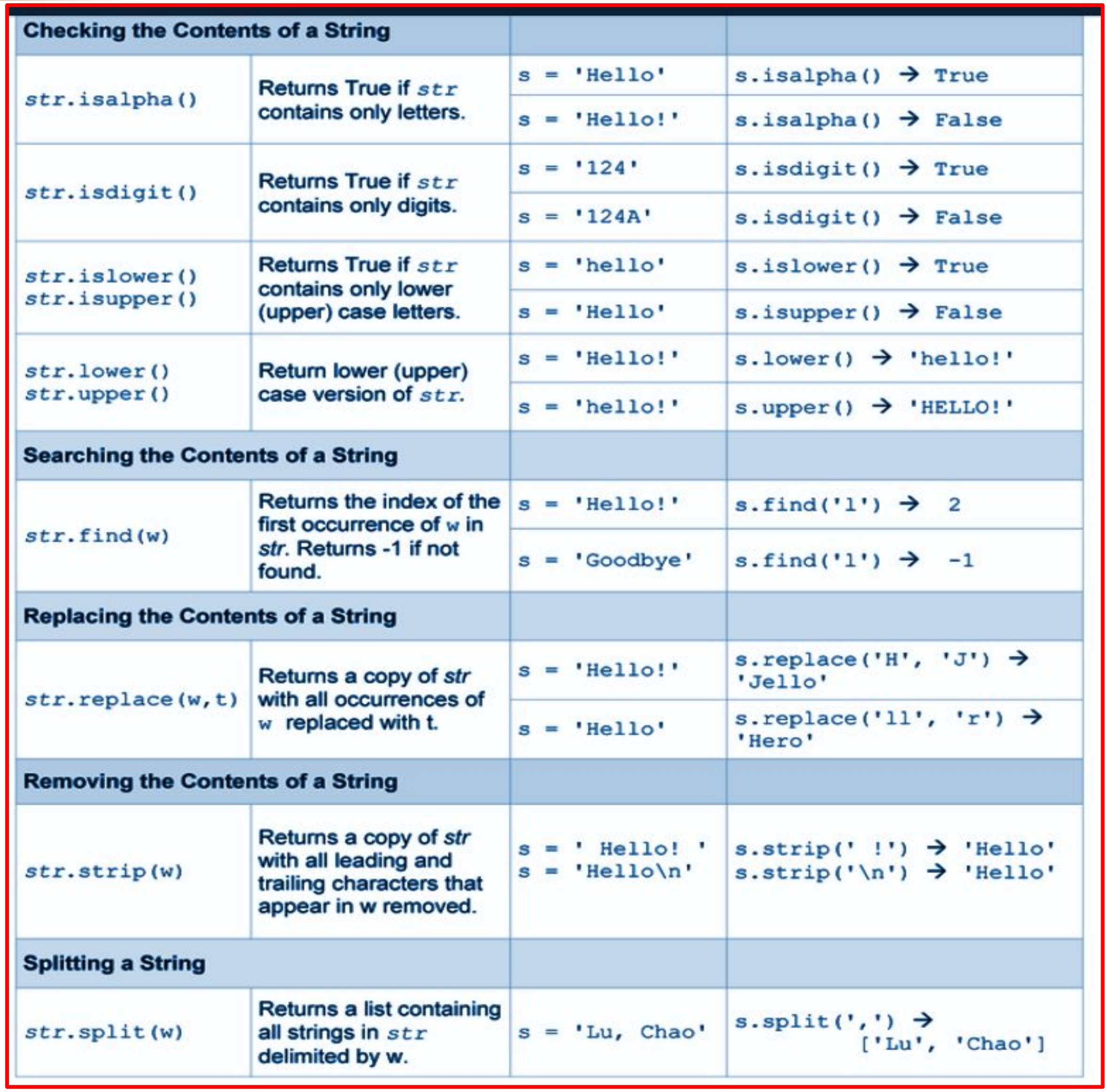


Sequences Operations Applicable to Strings			
Length	`len(str)`	Membership	`'h' in s`
Select	`s[index_val]`	Concatenation	`s + w`
Slice	`s[start:end]`	Minimum Value	`min(s)`
Count	`s.count(char)`	Maximum Value	`max(s)`
Index	`s.index(char)`	Comparison	`s == w`

pCPS 8.3.3 String Methods in python

- There are a **number** of methods specific to strings in addition to the general sequence operations

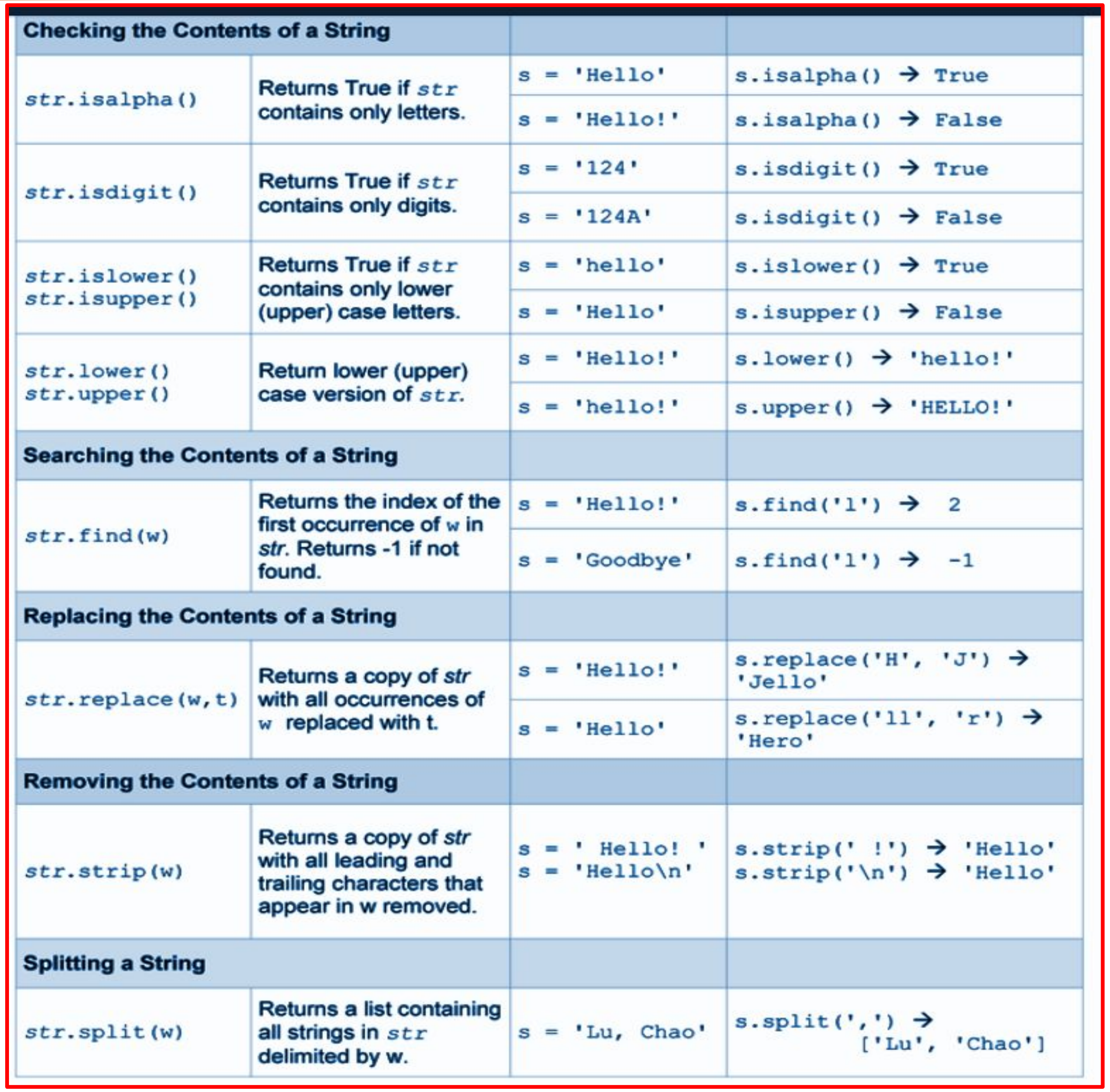
- There are **times** when the **individual characters** in a **string** or **substring** needs to be checked.



Checking the Contents of a String			
`str.isalpha()`	Returns True if `str` contains only letters.	`s = 'Hello'`	`s.isalpha()` → `True`
		`s = 'Hello!'`	`s.isalpha()` → `False`
`str.isdigit()`	Returns True if `str` contains only digits.	`s = '124'`	`s.isdigit()` → `True`
		`s = '124A'`	`s.isdigit()` → `False`
`str.islower()` `str.isupper()`	Returns True if `str` contains only lower (upper) case letters.	`s = 'hello'`	`s.islower()` → `True`
		`s = 'Hello'`	`s.isupper()` → `False`
`str.lower()` `str.upper()`	Return lower (upper) case version of `str`.	`s = 'Hello!'`	`s.lower()` → `'hello!'`
		`s = 'hello!'`	`s.upper()` → `'HELLO!'`
Searching the Contents of a String			
`str.find(w)`	Returns the index of the first occurrence of `w` in *str*. Returns -1 if not found.	`s = 'Hello!'`	`s.find('l')` → 2
		`s = 'Goodbye'`	`s.find('l')` → -1
Replacing the Contents of a String			
`str.replace(w,t)`	Returns a copy of *str* with all occurrences of `w` replaced with t.	`s = 'Hello!'`	`s.replace('H', 'J')` → `'Jello'`
		`s = 'Hello'`	`s.replace('ll', 'r')` → `'Hero'`
Removing the Contents of a String			
`str.strip(w)`	Returns a copy of *str* with all leading and trailing characters that appear in w removed.	`s = ' Hello! '` `s = 'Hello\n'`	`s.strip(' !')` → `'Hello'` `s.strip('\n')` → `'Hello'`
Splitting a String			
`str.split(w)`	Returns a list containing all strings in `str` delimited by w.	`s = 'Lu, Chao'`	`s.split(',')` → `['Lu', 'Chao']`

pCPS 8.3.3 String Methods in python

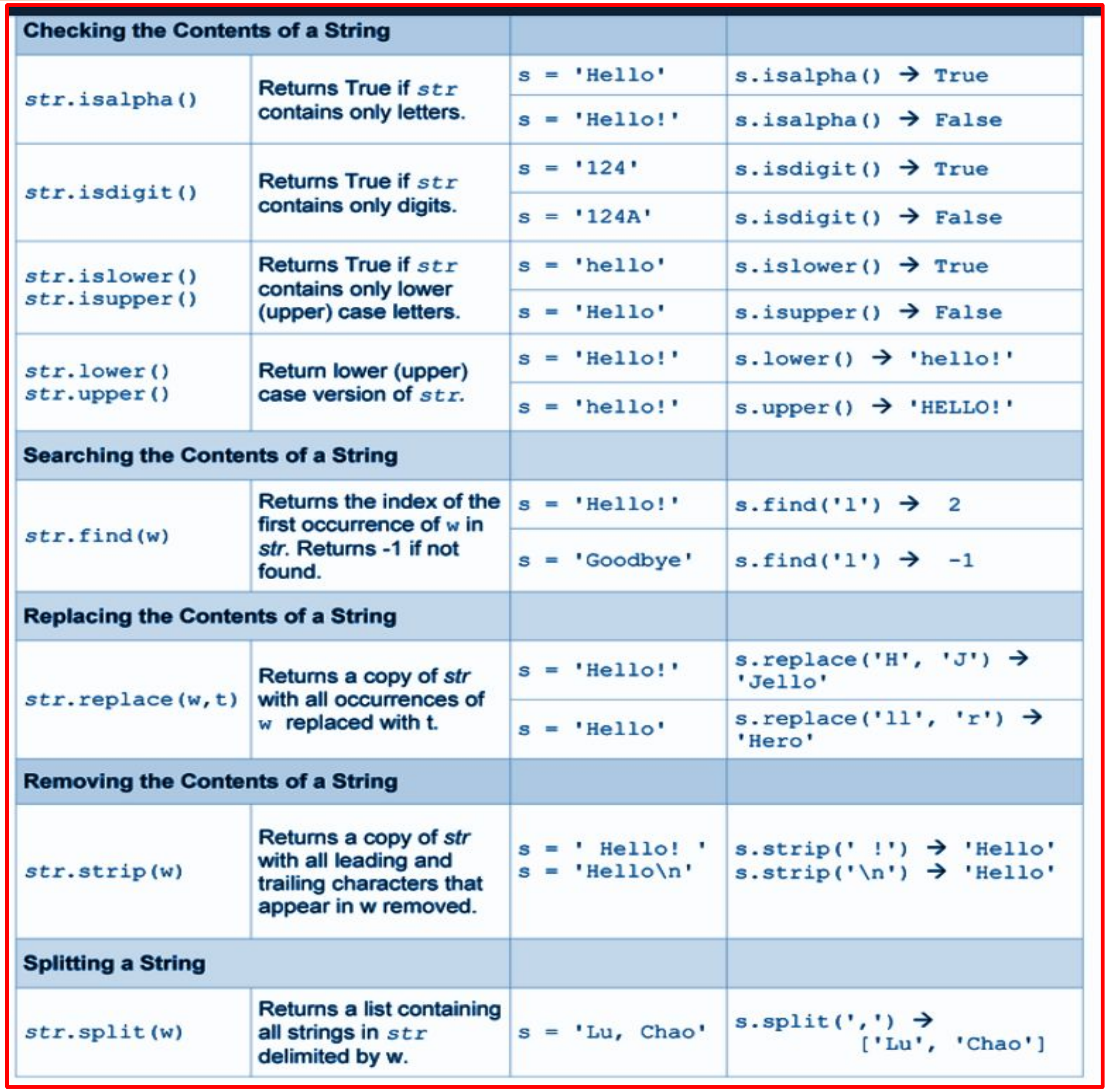
- String processing involves **search**.
- The **find** method returns the **index location** of the **first** occurrence of a specified **substring**.
- Since in python strings are **immutable**, to **update** the string, a **new** string **would** be **constructed** with the desired replacement
- The **replace** method **produces** a **new string** with **every occurrence** of a **given substring** within the **original** string replaced with another
- Note that for all **string modifications**, the variable **references** the **same** string until it is **reassigned**.



Checking the Contents of a String			
`str.isalpha()`	Returns True if *str* contains only letters.	`s = 'Hello'`	`s.isalpha() → True`
		`s = 'Hello!'`	`s.isalpha() → False`
`str.isdigit()`	Returns True if *str* contains only digits.	`s = '124'`	`s.isdigit() → True`
		`s = '124A'`	`s.isdigit() → False`
`str.islower()` `str.isupper()`	Returns True if *str* contains only lower (upper) case letters.	`s = 'hello'`	`s.islower() → True`
		`s = 'Hello'`	`s.isupper() → False`
`str.lower()` `str.upper()`	Return lower (upper) case version of *str*.	`s = 'Hello!'`	`s.lower() → 'hello!'`
		`s = 'hello!'`	`s.upper() → 'HELLO!'`
Searching the Contents of a String			
`str.find(w)`	Returns the index of the first occurrence of w in *str*. Returns -1 if not found.	`s = 'Hello!'`	`s.find('l') → 2`
		`s = 'Goodbye'`	`s.find('l') → -1`
Replacing the Contents of a String			
`str.replace(w,t)`	Returns a copy of *str* with all occurrences of w replaced with t.	`s = 'Hello!'`	`s.replace('H', 'J') → 'Jello'`
		`s = 'Hello'`	`s.replace('ll', 'r') → 'Hero'`
Removing the Contents of a String			
`str.strip(w)`	Returns a copy of *str* with all leading and trailing characters that appear in w removed.	`s = ' Hello! '` `s = 'Hello\n'`	`s.strip(' !') → 'Hello'` `s.strip('\n') → 'Hello'`
Splitting a String			
`str.split(w)`	Returns a list containing all strings in *str* delimited by w.	`s = 'Lu, Chao'`	`s.split(',') → ['Lu', 'Chao']`

pCPS 8.3.3 String Methods in python

- String processing involves **search**.
- python provides a **strip** method that “**strips off**” **leading** and **trailing** characters from a **string**.
- **strip** method is especially **useful** for **stripping** off the **newline character**, \n, from the **end** of a **line** in text processing if needed



Checking the Contents of a String			
`str.isalpha()`	Returns True if `str` contains only letters.	`s = 'Hello'`	`s.isalpha() → True`
		`s = 'Hello!'`	`s.isalpha() → False`
`str.isdigit()`	Returns True if `str` contains only digits.	`s = '124'`	`s.isdigit() → True`
		`s = '124A'`	`s.isdigit() → False`
`str.islower()` `str.isupper()`	Returns True if `str` contains only lower (upper) case letters.	`s = 'hello'`	`s.islower() → True`
		`s = 'Hello'`	`s.isupper() → False`
`str.lower()` `str.upper()`	Return lower (upper) case version of `str`.	`s = 'Hello!'`	`s.lower() → 'hello!'`
		`s = 'hello!'`	`s.upper() → 'HELLO!'`
Searching the Contents of a String			
`str.find(w)`	Returns the index of the first occurrence of `w` in *str*. Returns -1 if not found.	`s = 'Hello!'`	`s.find('l') → 2`
		`s = 'Goodbye'`	`s.find('l') → -1`
Replacing the Contents of a String			
`str.replace(w,t)`	Returns a copy of *str* with all occurrences of `w` replaced with t.	`s = 'Hello!'`	`s.replace('H', 'J') → 'Jello'`
		`s = 'Hello'`	`s.replace('ll', 'r') → 'Hero'`
Removing the Contents of a String			
`str.strip(w)`	Returns a copy of *str* with all leading and trailing characters that appear in w removed.	`s = ' Hello! '` `s = 'Hello\n'`	`s.strip(' !') → 'Hello'` `s.strip('\n') → 'Hello'`
Splitting a String			
`str.split(w)`	Returns a list containing all strings in `str` delimited by w.	`s = 'Lu, Chao'`	`s.split(',') → ['Lu', 'Chao']`

pCPS 8.3.3 String Methods in python

- The **title()** method returns a string where the first character in every word is upper case
- The **partition()** method searches for a specified string, and **splits** the string into a **tuple** containing **three elements**.
    - The **first** element contains the part **before** the specified string.
    - The **second** element contains the **specified** string.
    - The **third** element contains the part **after** the string.
- The **join()** method takes all **items** in an **iterable** and **joins** them into **one** string.
    - A string must be specified as the separator.
- The **capitalize()** method returns a string where the **first** character is **upper case**, and the **rest** is **lower case**.
- The **isnumeric()** method **returns True** if **all** the **characters** are **numeric** (0-9), else **False**.

THANK YOU

Nitin V Pujari
Faculty, Computer Science
Dean - IQAC, PES University
nitin.pujari@pes.edu

For Course Digital Deliverables visit [**www.pesuacademy.com**](http://www.pesuacademy.com)